Amendments to the Claims:

Listing of Claims:

Listing of Claims:

1-4. (Canceled)

5. (Currently Amended) A method in a data processing system for gathering performance information associated with executing instructions, the method comprising:

compiling source code statements, by a compiler, to generate instructions for a processor in the data processing system;

generating, by the compiler, indicators to be processed by the processor during execution of the instructions, wherein the indicators are data values that are retrieved from memory during execution of the instructions [[that]] and specify counting of events that are associated with execution of the instructions or are data values that are retrieved from memory during execution of the instructions [[that]] and specify counting of events that are associated with accesses to memory locations, and wherein the indicators specify counting of events on a per instruction or a per memory location basis such that only the events specified by the indicators are counted to form the performance information; and

generating instructions to select an execution path within a set of instructions based on a count value in the performance information that represents counted events, wherein a branch instruction that selects the execution path is contained in an object code block that contains the instructions that are executed while the indicators are processed.

6. (Previously Presented) The method of claim 5 further comprising:

processing, by the compiler, source code statements that control the selection of the execution path.

7. (Previously Presented) The method of claim 5 further comprising:

determining, by the compiler, alternative ways to compile the source code statements;

compiling the source code statements in multiple ways to generate multiple sets of instructions for the compiled source code statements;

and placing each set of instructions in a different execution path to be selected based on the count value.

8. (Original) The method of claim 7 further comprising:

identifying a compiler directive among the source code statements that specifies a type of autonomic monitoring to be implemented by the compiler; and

identifying alternative ways to compile the source code statements based on the compiler directive.

9. (Original) The method of claim 8 further comprising:

specifying an event in the compiler directive to be autonomically monitored.

10. (Previously Presented) The method of claim 5 further comprising:

determining, by the compiler, alternative ways to compile a subroutine within the source code statements;

generating a plurality of compiled versions of the subroutine, wherein each compiled version has been compiled differently from other compiled versions of the subroutine; and

placing the plurality of compiled versions of the subroutine in different execution paths to be selected based on the count value.

11. (Currently Amended) The method of claim 5 further comprising:

processing, by the compiler, at least one compiler directive among the source code statements that specifies to the compiler the selection of the execution path.

12-15. (Canceled)

16. (Currently Amended) A computer program product on a computer readable recordable – type storage medium for use in a data processing system that executes instructions, the computer program product comprising:

instructions for compiling source code statements to generate instructions for a processor in the data processing system;

instructions for generating, by the compiler, indicators to be processed, by the processor, during execution of the instructions, wherein the indicators are data values that are retrieved from memory during execution of the instructions [[that]] and specify counting of events that are associated with execution of the instructions or are data values that are retrieved from memory during execution of the instructions [[that]] and specify counting of events that are associated with accesses to memory locations, and wherein the indicators specify counting of events on a per instruction or a per memory location basis such that only the events specified by the indicators are counted; and

instructions for generating instructions to select an execution path within a set of instructions based on a count value that represents counted events, wherein a branch instruction that selects the execution path is contained in an object code block that contains the instructions that are executed while the indicators are processed.

17. (Previously Presented) The computer program product of claim 16 further comprising:

instructions for processing, by the compiler, source code statements that control the selection of the execution path.

18. (Previously Presented) The computer program product of claim 16 further comprising:

instructions for determining, by the compiler, alternative ways to compile the source code statements;

instructions for compiling the source code statements in multiple ways to generate multiple sets of instructions for the compiled source code statements; and

instructions for placing each set of instructions in a different execution path to be selected based on the count value.

19. (Previously Presented) The computer program product of claim 18 further comprising:

instructions for identifying a compiler directive among the source code statements that specifies a type of autonomic monitoring to be implemented by the compiler; and instructions for identifying alternative ways to compile the source code statements based on the compiler directive.

20. (Previously Presented) The computer program product of claim 19 further comprising:

instructions for specifying an event in the compiler directive to be autonomically monitored.

21. (Previously Presented) The computer program product of claim 16 further comprising:

instructions for determining, by the compiler, alternative ways to compile a subroutine within the source code statements;

instructions for generating a plurality of compiled versions of the subroutine, wherein each compiled version has been compiled differently from other compiled versions of the subroutine; and

instructions for placing the plurality of compiled versions of the subroutine in different execution paths to be selected based on the count value.

22. (Currently Amended) The computer program product of claim 16 further comprising: instructions for processing, by the compiler, compiler directives among the source code statements that specify the selection of the execution path.

23-26. (Canceled)

27. (Currently Amended) An apparatus in a data processing system that executes instructions, the apparatus comprising:

a compiler, wherein the compiler compiles source code statements to generate instructions for a processor in the data processing system generates indicators to be processed during execution of the instructions, wherein the indicators are data values that are retrieved from memory during execution of the instructions [[that]] and specify counting of events that are associated with execution of the instructions or are data values that are retrieved from memory during execution of the instructions [[that]] and specify counting of events that are associated with accesses to memory location and wherein the indicators specify counting of events on a per instruction or a per memory location basis such that only the events specified by the indicators are counted to form the performance information; and generates instructions to select an execution path within a set of instructions based on a count value that represents counted events, wherein a branch instruction that selects the execution path is contained in an object code block that contains the instructions that are executed while the indicators are processed.

28. (Previously Presented) The apparatus of claim 27 wherein the compiler processes source code statements that control the selection of the execution path.

29. (Previously Presented) The apparatus of claim 27 further comprising: means for determining, by the compiler, alternative ways to compile the source code statements;

means for compiling the source code statements in multiple ways to generate multiple sets of instructions for the compiled source code statements; and

means for placing each set of instructions in a different execution path to be selected based on the count value.

30. (Original) The apparatus of claim 29 further comprising:

means for identifying a compiler directive among the source code statements that specifies a type of autonomic monitoring to be implemented by the compiler; and

means for identifying alternative ways to compile the source code statements based on the compiler directive.

31. (Original) The apparatus of claim 30 further comprising:

means for specifying an event in the compiler directive to be autonomically monitored.

32. (Previously Presented) The apparatus of claim 27 further comprising:

means for determining, by the compiler, alternative ways to compile a subroutine within the source code statements;

means for generating a plurality of compiled versions of the subroutine, wherein each compiled version has been compiled differently from other compiled versions of the subroutine; and means for placing the plurality of compiled versions of the subroutine in different execution paths to be selected based on the count value.

33. (Currently Amended) The apparatus of claim 27 further comprising:

means for processing, by the compiler, at least one compiler directive among the source code statements that specifies the selection of the execution path.

34. (Newly Added) The method of claim 5, further comprising:

determining whether a given instruction is associated with one of the indicators.

35. (Newly Added) The method of claim 34, further comprising:

responsive to determining that a given instruction is associated with one of the indicators, sending a signal to a performance monitoring unit of the processor; and

responsive to the performance monitoring unit receiving the signal, the performance monitoring unit counting events associated with the given instruction.

36. (Newly Added) The method of claim 35, wherein the determining and sending steps are performed by a cache of the processor.